------------------------- CEN 6070 Software Testing & Verification -------------------------

Exam 2 – Summer 2016 – Solution Notes

1. a. would, b. would not, c. would not, d. would, e. would not, f. would not, g. would, h. would not

2. a. true, b. false, c. false, d.-i. true

3. a.

$$\frac{\mathbf{P \Rightarrow wlp(S,Q)}}{\{P\}\ S\ \{Q\}}$$

b. H_1: $wp(s,\ c \wedge Q)$
H_2: $wp(s,\ \neg c \wedge H_1)$
H_3: $wp(s,\ \neg c \wedge H_2)$
H_k: $wp(s,\ \neg c \wedge H_{k-1})$

c. H_1: $wp(S,\ t=0 \wedge z=yx) = \mathbf{(\ t=1 \wedge z=y(x-1)\ )}$
H_2: $wp(S,\ t\neq 0 \wedge t=1 \wedge z=y(x-1)) = (t\neq 1 \wedge t=2 \wedge z=y(x-2))$
$\qquad = \mathbf{(\ t=2 \wedge z=y(x-2)\ )}$
H_3: $\mathbf{(t=3 \wedge z=y(x-3))}$
H_k: $\mathbf{t=k \wedge z=y(x-k))}$

Closed form expression of **wlp**: $\mathbf{(t>0 \wedge z=y(x-t))\ \vee\ t\leq 0}\ \equiv\ \mathbf{z=y(x-t)\ \vee\ t\leq 0}$

d. To prove the given assertion using the ROI from part (a), we need to show that:
$(t=2 \wedge x=6 \wedge z=8 \wedge y=2) \Rightarrow wlp(K,\ z=yx)$
That is, we need to show:
$(t=2 \wedge x=6 \wedge z=8 \wedge y=2) \Rightarrow (t>0 \wedge z=y(x-t))\ \vee\ t\leq 0$
Since $(t=2 \wedge x=6 \wedge z=8 \wedge y=2) \Rightarrow (\ wlp(K,\ z=yx) = (2>0 \wedge 8=2(6-2))\ \vee\ 2\leq 0$
$= \text{True}\ )$,
the assertion holds.

4. p1 = (x>1 -> x, z := 1, zx!)
p2 = (x>2 -> x, z := 2, z(x-1)! | true -> x, z := x-1, z(x-1))

	P1	*P2*
f1	S	N
f2	N	C

5. a. valid, b-e invalid, f valid, g.-h. invalid, i. valid, j. invalid

6. **INITIALIZATION: Does {P} s {I}?**

P: $\{\text{true} \land z=0 \land t=x\}$

$z := z+y$

$\{z=y \land t=x\}$

$t := t-1$

$\{z=y \land t+1=x\} \Rightarrow z=y(x-t) \equiv I$ √

PRESERVATION: Does {I ∧ ¬b} s {I}?

$I \land \neg b$: $\{ z=y(x-t) \land t\neq 0 \}$

$z := z+y$

$\{ z=y(x-t+1) \land t\neq 0 \}$

$t := t-1$

$\{ z=y(x-(t+1)+1) \land t+1\neq 0 \}$

$\Rightarrow$

$\{ z=y(x-t) \} \equiv I$ √

FINALIZATION: Does (I ∧ b) ⇒ Q?

$(I \land b)$: $(z=y(x-t) \land t=0) \Rightarrow z=yx \equiv Q$ √

7. **Does term*(f, H)?***

(We use the Method of Well Founded Sets with measure k to prove *H* will terminate for any initial (integer) value of t>0.)

i. the value of t decreases by 1 with each execution of the loop body (via t := t-1).
ii. the value of t is bounded from below when t is initially greater than 0 since when t becomes equal to 0, the loop must terminate because "t=0" (i.e., the loop termination predicate) becomes true.
iii. the value of t may assume only a finite number of values $[(t_0>0, t_0-1, t_0-2, \ldots, 0)]$ since it decreases by an integral amount (1) with each iteration of the loop body.

Therefore, *H* terminates for any initial value of t>0 and we conclude that term*(f, H) holds.*

Does *(p o g)* ⇒ *(f = g)*?

$[\ (t=0)\ o\ (t,z := t-1, z+y)\ ] \Rightarrow (t_0=1)$

$(t=1) \Rightarrow (\ f = (t,z := 0, z+y(1))$

$= (t,z := 0, z+y)\ )$

$(t=1) \Rightarrow (\ g = (t,k := 1-1, z+y)\ ).$

$= (t,z := 0, z+y)\ )$

Therefore, *(p o g) ⇒ (f = g)*.

7. (cont'd)

Does $\neg(p\ o\ g) \Rightarrow (f = f\ o\ g)$?

$\neg$ $[\ (t=0)\ o\ (t,z := t-1,z+y)\ ] \Rightarrow (t_0 \neq 1)$

Thus, there are 2 cases to consider: $t_0<1$ and $t_0>1$.

case a:

$(t>1) \Rightarrow (\ f = (t,z := 0,z+yt)\ )$

$(t>1) \Rightarrow (\ f \text{ o } g\ =$ **$(t,z := 0,z+yt)$** o

$(t,z := t-1,z+y)$

since $((t>0)$ o $g(t>1)) =$ true

$= (t,z := 0,(z+y)+y(t-1))$

$= (t,z := 0,(z+y+yt-y))$

$= (t,z := 0,(z+yt))\ )$

case b:

$(t<1) \Rightarrow (\ f =$ undefined)

$(t<1) \Rightarrow (\ f \text{ o } g\ =$ **undefined** o g

since $((t>0)$ o $g(t<1)) =$ false

= undefined)

Therefore, $\neg(p\ o\ g) \Rightarrow (f = f\ o\ g)$.

8. a. For a loop computing a function, *f,* the function value of the current state, X, is the same as the function value of the initial state, X_0. That is: $f(X)=f(X_0)$. This relationship is commonly represented as $q(\mathbf{X}) = (\ f(\mathbf{X})=f(X_0)\ )$ and is an *f* adequate "**invariant**" of any loop computing *f.*

   b. The derivation of $q(X)$ requires knowledge of the program function, *f.* Furthermore, if the **specified** post-condition, Q, is not of the form $X=f(X_0)$, the translation between Q and *f* may not be obvious.

9. a. Yes, the given invariants can be used with the two given "output" values to solve for any common initial or intermediate state (x_0, y_0). By substituting x=0, y=7 in $q_t(X)$ and x=-2, y=-9 in $q_s(X)$, we obtain simultaneous equations: $7=y_0+2x_0$ and $-3=y_0-3x_0$ which are satisfied by $x_0=2$ and $y_0=3$. Thus, (2,3) is a common initial or intermediate state for both loops.

10. a. true, b. false, c.-e. true

11. In *statistical usage testing,* test cases are derived from usage probability distributions reflecting external system behavior – not internals of design and implementation as in conventional *coverage testing*. In effect, *statistical usage testing* amounts to testing software the way users are expected to use it. Thus errors tend to be found in failure-rate order (finding higher-rate errors sooner) on average, whereas traditional *coverage testing* finds errors in random order – i.e., in no particular rate order. Finding errors in failure-rate order results in dramatic improvements in MTTF (Mean Time To Failure – a popular measure of software reliability) over random order.

Histogram of Raw Scores

```
                                                                    *
          *                                     *       *         * *     *
1...222222222233333333334444444444555555555566666666667777777777888888888899999999991111111111111111111111111111111…1
    012345678901234567890123456789012345678901234567890123456789012345678901234567890000000000111111111122222222223…6
                                                                                    0123456789012345678901234567890…3
                                                        ^
                                                        |
                                                       AVG
```